Show all of your work clearly in the space provided or on the additional page at the end of the exam. If the additional page is used, clearly identify to which exam question it is related. Be sure to **read each problem carefully**. Note that the exam is double sided.

1. (20 points) True/False (**T** or **F**)

_____ A mutator method must change the value of an attribute within a class.

_____ Math.PI is declared as a `private` attribute.

_____ It is not possible to tell if a method is `private` or `public` by just looking at the UML class diagram.

_____ Math.cos(30) is a call to a class method.

_____ Local variables declared in a constructor are only accessible within that constructor.

_____ A `private` attribute is accessible to all methods defined within the class.

_____ An accessor method must be declared as `static`.

_____ It is illegal to place `this` on the left side of the assignment operator. E.g.,

```
this = that;
```

_____ All classes in a Java package must be stored in the same folder/directory.

_____ A class may have two attributes with exactly the same name as long as they are from different types and one of them is declared with this. in front of it. E.g.,

```
private double number;
private int this.number;
```

2. (5 points) Write one line of code that declares a class constant that represents the number of seconds in a minute.

3. (10 points) Explain the purpose of the `import` statement.

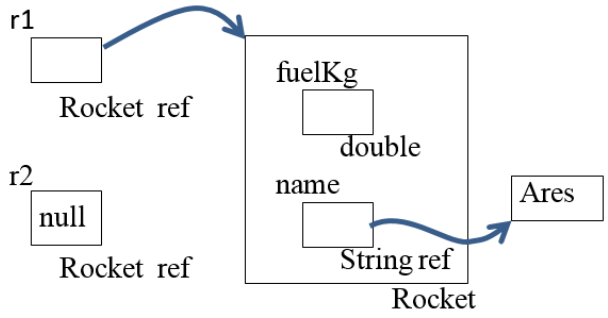
4. (10 points) Describe the role of the reserved word `this`.

5. (15 points) Suppose a Rocket class exists with two attributes: fuelKg (a double) and name (a String). Complete the following diagram that illustrates the state of memory at four points in the program on the right. The solution to step **(b)** is provided as an example.

(a)

```
Rocket r1 = null;
Rocket r2 = null;
// (a)
r1 = new Rocket("Ares");
// (b)
r1.setFuelKg(5000);
r2 = r1;
// (c)
r1 = new Rocket("Saturn IV");
// (d)
```

(b)



(c)

(d)

6. (15 points) Consider the following code:

```
public static void main(String[] args) {
    Timer t1 = Timer();
    System.out.println("Timer 1: " + t1.display());
    Timer t2 = Timer(15);
    System.out.println("Timer 2: " + t2.getMinutes() + " minutes");
    Timer t3 = t1.plus(t2);
    System.out.println("Timer 3: " + t3.display());
    t1.clear();
    System.out.println("Timer 1: " + t1.display());
    t2.addMinutes(1);
    System.out.println("Timer 2: " + t2.display());
    t3.addSeconds(25);
    System.out.println("Timer 3: " + t3.display());
    t3.tick();
    t3.tick();
    t3.tick();
    System.out.println("Timer 3: " + t3.display());
}
```

Create a UML class diagram for the Timer class used in the above code that, if implemented, would allow the main method to produce the following output:

```
Timer 1: 60 seconds
Timer 2: 0.25 minutes
Timer 3: 75 seconds
Timer 1: 0 seconds
Timer 2: 75 seconds
Timer 3: 100 seconds
Timer 3: 97 seconds
```

7. (25 points) Completely implement the Timer class from the previous problem. You implementation should allow the main method to compile without errors and produce the sample output shown.

Additional space — indentify which problem your work is associated with.